TUGAS PROJEK UTS
Time Out pada Sensor

I. Tujuan :

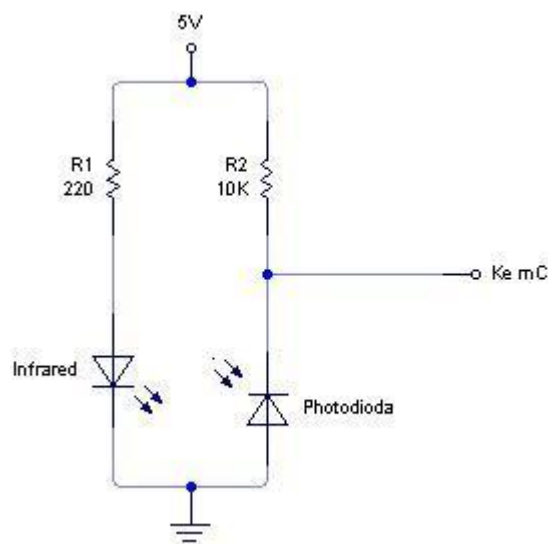
- Mahasiswa memahami tentang program time out dan bouncing.
- Mahasiswa memahami kerja sensor dengan time out dan bouncing.
- Mahasiswa mampu memprogram modul STM32 menggunakan STM32CubeMX dan Eclipse AC6 dengan gaya pemrogaman berstruktur.
- Mahasiswa mampu membuat GUI (Graphical User Interface) untuk sistem embedded yang dibuat.
- Mahasiswa mampu menggunakan dan mengkonfigurasi fitur yang ada pada modul STM32 seperti Timer, USART, dan GPIO.

II. Teori Penunjang

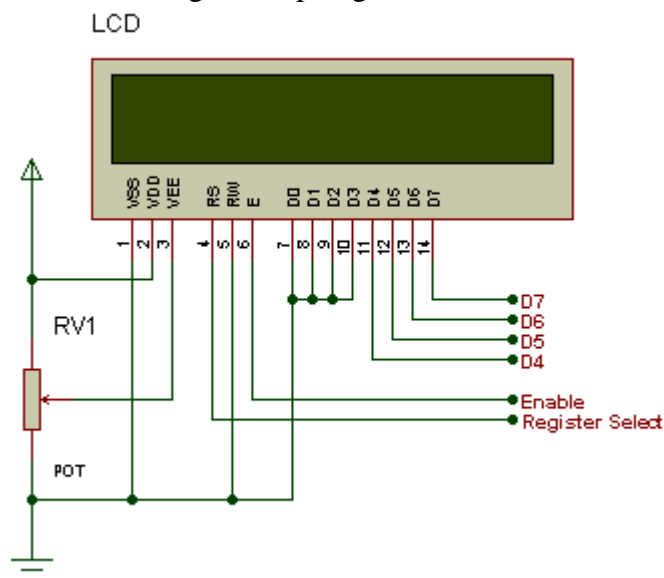
1. Mikrokontroller arm

   Arsitektur **ARM** merupakan arsitektur prosesor 32-bit RISC yang dikembangkan oleh *ARM Limited*. Dikenal sebagai *Advanced RISC Machine* di mana sebelumnya dikenal sebagai *Acorn RISC Machine*. Pada awalnya merupakan *prosesor desktop* yang sekarang didominasi oleh keluarga x86. Namun desain yang sederhana membuat prosesor ARM cocok untuk aplikasi berdaya rendah. Hal ini membuat prosesor ARM mendominasi pasar *mobile electronic* dan *embedded system* di mana membutuhkan daya dan harga yang rendah.

   Pada tahun 2007, sekitar 98% dari satu miliar *mobile phone* yang terjual menggunakan setidaknya satu buah prosesor ARM. Dan pada tahun 2009, prosesor ARM mendominasi sekitar 90% dari keseluruhan pasar prosesor 32-bit RISC. Prosesor ARM digunakan di berbagai bidang seperti elektronik umum, termasuk PDA, mobile phone, media player, music player, game console genggam, kalkulator dan periperal komputer seperti hard disk drive dan router.

2. Sensor photodioda

3. Konfigurasi LCD 16X2

   Display elektronik adalah salah satu komponen elektronika yang berfungsi sebagai tampilan suatu data, baik karakter, huruf ataupun grafik. LCD (Liquid Cristal Display) adalah salah satu jenis display elektronik yang dibuat dengan teknologi CMOS logic yang bekerja dengan tidak menghasilkan cahaya tetapi memantulkan

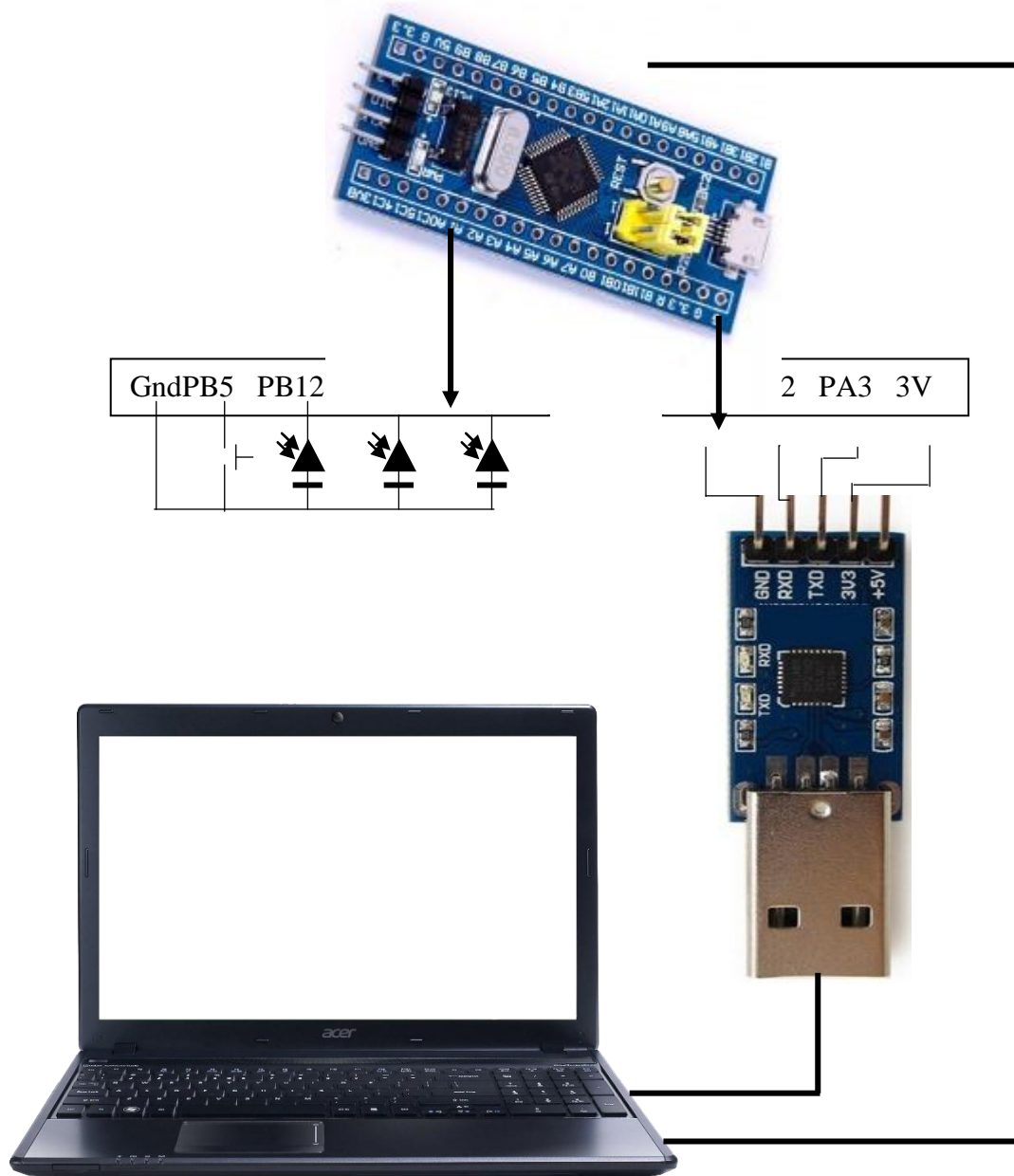
cahaya yang ada di sekelilingnya terhadap front-lit atau mentransmisikan cahaya dari back-lit. LCD (Liquid Cristal Display) berfungsi sebagai penampil data baik dalam bentuk karakter, huruf, angka ataupun grafik.

4. **Timer and Counter**

Timer & Counter merupakan fitur yang telah tertanam di mikrokontroler A yang memiliki fungsi terhadap waktu. Fungsi pewaktu yang dimaksud disini adalah penentuan kapan program tersebut dijalankan, tidak hanya itu saja fungsi timer yang lainnya adalah PWM, ADC, dan Oscillator. Prinsip kerja timer dengan cara membagi frekuensi (prescaler) pada clock yang terdapat pada mikrokontroler sehingga timer dapat berjalan sesuai dengan frekuensi yang di kehendaki.

Timer merupakan fungsi waktu yang sumber clocknya berasal dari clock internal. Sedangkan counter merupakan fungsi perhitungan yang sumber clocknya berasal dari external mikrokontroler. Salah satu contoh penggunaan fungsi timer yaitu pada jam digital yang sumber clocknya bisa menggunakan crystal oscillator dan contoh penggunaan counter pada penghitung barang pada konveyor yang sumber clocknya berasal dari sensor yang mendeteksi barang tersebut.

III. Alat dan Bahan

No.	Alat	**No.**	Bahan
1.	Solder dan Timah	1.	ARM STM32F1
2.	Tang	2.	LCD 16x2
3.	Gunting	3.	Board Rangkaian
4.	Penggaris	4.	Switch
5.	Pemotong	5.	Power Bank
6.	Lem	6.	Kabel
7.	Obeng	7.	Komponen Pendukung
8.	Avometer	8.	Sensor Cahaya dengan Komparator
9.	Bor	9.	Mobil Tamiya

IV. Rangkaian Sistem

V. Langkah Kerja

1. Mendesain Rangkaian dan Board pada software EAGLE.
2. Mencetak Board pada PCB
3. Menyiapkan peralatan dan komponen yang diperlukan.
4. Menyolder komponen – komponen ke PCB sesuai dengan gambar rangkaian sistem.
5. Setelah selesai menyolder dan troubleshoot rangkaian, board di *packaging* sesuai aturan *The Makers of Bill.*
6. Menginstall aplikasi dan driver untuk memprogram STM32F1 Discovery yaitu : driver ST-Link, driver USB Prolific, aplikasi STM32CubeMX, dan aplikasi Eclipse AC6.
7. Memasang modul STM32F! ke *shield* buatan pada PCB kemudian menghubungkannya ke laptop menggunakan kabel mini USB.
8. Membuka aplikasi STM32CubeMX dan terlebih dahulu menginstall library F1.

9. Membuat project baru pada aplikasi STM32CubeMX untuk board STM32F1 lalu mengatur konfigurasi fitur - fitur yang akan digunakan yaitu : Clock RCC, Timer2, USART, GPIO.
10. Men-save project dan meng-*generate* project kedalam bentuk file – file program.
11. Membuka aplikasi Eclipse AC6 (System Workbench for STM32), kemudian meng-*import* project yang sebelumnya sudah digenerate dari STM32CubeMX.
12. Melakukan pemrogaman pada Eclipse AC6 untuk sistem embedded yang akan dibuat yaitu "Sistem embedded untuk balapan mobil" dengan gaya pemrogaman berstruktur.
13. Mengcompile program kemudian mendownloadkannya ke STM32F1.
14. Menghubungkan pin Serial dari STM32 ke modul Bluetooth.
15. Membuka situs Mit App Inventor 2 untuk membuat aplikasi monitoring andriod.
16. Mendesain GUI sesuai dengan gambar desain GUI yang terlampir.
17. Memprogram aplikasi GUI untuk memilah – milah data yang dikirimkan oleh STM32F1.
18. Melakukan debugging aplikasi GUI yang telah dibuat.
19. Menghubungkan bluetooth modul dengan bluetooth andriod.
20. Melakukan uji coba sistem dengan menerangi sensor photodiode kemudian secara bergantian menutupi sensor untuk melihat respon sistem embedded yang dibuat.
21. Hasil projek dapat dilihat pada lampiran hasil projek.

VI. ANALISA DATA

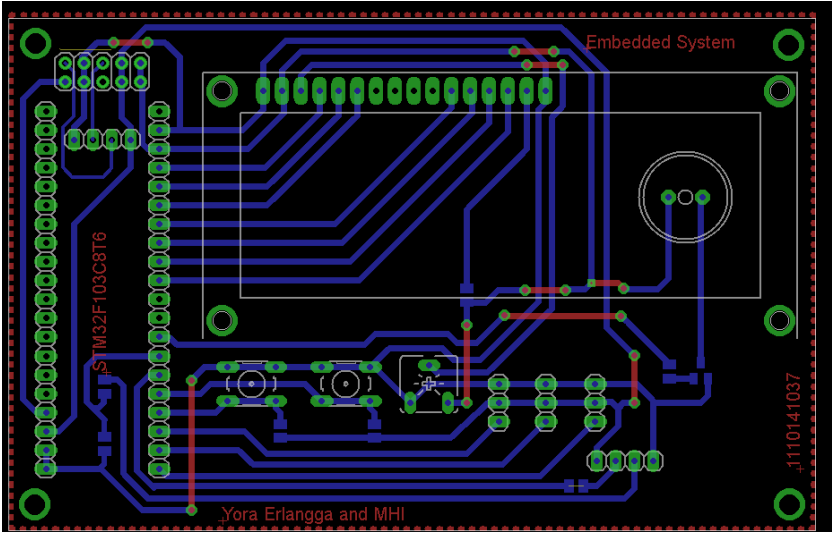
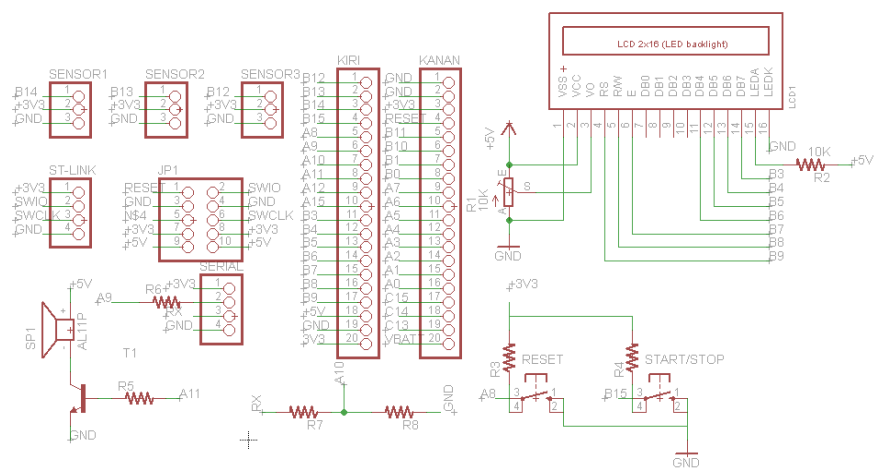
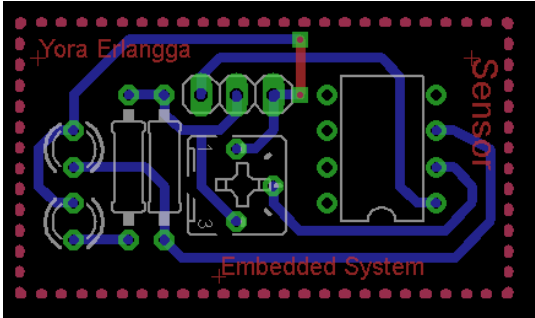
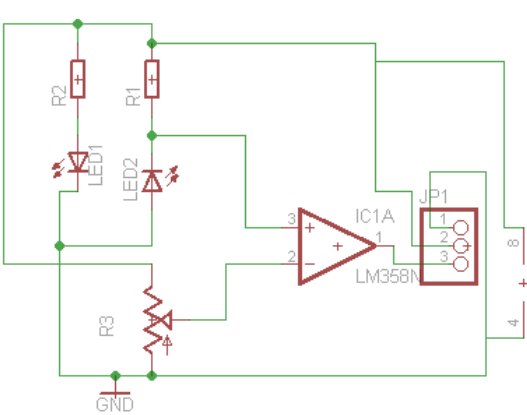
Pada tugas projek kali ini yaitu tentang aplikasi time out pada sensor. Projek ini di aplikasikan pada sebuah Embedded Multi Stopwatch digunakan untuk merekam perlombaan mobil dengan jumlah lap sebanyak 4 Lap dan di ikuti 3 Kendaraan. Pembangkitan timer dilakukan oleh Embedded Controller, sementara tampilan jarak tempuh untuk setiap lap ditampilkan di LCD. Agar tidak terjadi bouncing pada saat penekanan tombol, maka ditambahkan mekanisme anti-bouncing menggunakan software. Pada projek ini mempunyai spesifikasi cara kerja sebagai berikut :

- Tombol start/stop digunakan untuk memulai atau menghentikan timer
- Tombol reset digunakan untuk mereset timer secara manual
- Sensor 1 sampai 3 digunakan untuk mencatat waktu yang ditempuh pada setiap lap (3 Sensor untuk 3 Kendaraan) dengan tambahan mekanisme anti bouncing menggunakan software
- Apabila sensor mengalami kerusakan maka sistem akan memberitahu user tentang nomer sensor yang mengalami kegagalan (kode pesan setiap sensor berbeda) dengan menggunakan mekanisme timeout. Sistem juga mengirimkan kode pesn error tersebut ke PC melalui serial untuk keperluan debugging.

Berdasarkan penjelasan di atas, Embedded Controller menggunakan ARM STM32F1, lalu LCD menggunakan LCD Karakter 16x2. Lalu untuk tombol start dan reset menggunakan switch omron dan sensor cahaya menggunakan photodiode. Berikut konfigurasi pin-pin input outpu pada Embedded Controller :

No.	Input / Output	Pin
1.	LCD 16x2	B3-B9
2.	Sensor 1	B12

3.	Sensor 2	A12
4.	Sensor 3	B14
5.	UART	A9, A10
6.	Start	B12
7.	Reset	A11

Berikut rangkaian projek yang dibuat menggunakan eagle.

Rangkaian dan Board Embedded Controller

Rangkaian dan Board Sensor

Pada project ini, hal pertama yang dilakukan adalah membuat hardware pada pcb. Kemudian setelah selesai dilakukan pembuatan hardware, dilakukan pengecekan pada setiap komponen yang telah disolder, salah satunya dengan melakukan pengecekan pada data yang dikeluarkan oleh photodioda. Apabila photodioda mendapatkan cahaya maka, tegangan yang dihasilkan akan menjadi 0 volt dan apabila tidak mendapatkan cahaya maka, sensor akan mengeluarkan teganan 3 volt, tegangan ini digunakan untuk inputan mikrokontroller. Kemudian dilakukan pengecekan pada push button, apakah sudah bekerja dengan baik dan pengecekan pada kaki-kaki header apakah sudah tersambung dengan benar apa tidak. Setelah hardware telah terpenuhi maka dilakukan proses pemrograman yang dilakukan dengan menggunakan program seperti pada program diatas.

Proses selanjutnya adalah pemrograman STM32F1 dengan menggunakan Eclipse AC5, dimana diprogram untuk .h dan .c. Program yang digunakan program timer, UART, LCD, Sensor dan harwareinit. Program timer digunakan untuk melakukan pembuatan pewaktuan dalam proses pertandingan mobil. Kemudian dilakukan pembuatan UART dengan uart2 untuk melakukan komunikasi serial kepada komputer dengan menggunakan baudrate sebesar 9600 agar sesuai dengan baudrate dari komputer 9600. Uart dilakukan untuk menampilkan data pewaktuan, lab yang dilalui mobil dan lain-lain pada serial monitor pada komputer. Setelah itu, dilakukan pembacaan sensor yang dilakukan dengan menggunakan mekanisme timeout. Mekanisme timeout yang dilakukan ini, diperlukan untuk mengetahui apa ada hardware yang error apa tidak mekanismenya adalah yang pertama menentukan inisialisasi timeout dan berapa lamanya waktu yang dibutuhkaan saat akan timeout, kemudian timeout dimasukkan kedalam program saat sesudah program penantian bouncing selama 5 detik. Saat sensor mendeteksi tidak ada bouncing dan dilakukan pengecekan kembali sensor masih mendeteksi object maka ditunggu sampai object meninggalkan sensor, apabila object telah lewat dan masih selama waktu belum timeout maka akan didefinisikan ada object yang lewat. Namun apabila object terdeteksi dan sudah melewati masa time out maka, serial akan menuliskan atau mengirimkan data berupa sensor error dan tidak akan melakukan counting karena, program akan kembali menuju flag = 0 sama seperti mekanisme sensor, tombol start/stop dan reset juga dilakuakan mekanisme yang sama, kemudian pada setiap sensor dilakukan pembuatan mekanisme bouncing pada setiap sensor dan tombol yang digunakan untuk membuat sensor tidak mengcounting berkali-kali pada setiap sensing dan penekanan tombol.

Untuk melakukan penampilan pada lcd digunakan dengan pemanggilan sub routin dari lcd, dan pada setiap data yang akan ditampilkan dilakukan pemanggilan sub routin lcd_string untuk menampilkan data string. Mekanisme ini memerlukan penyimpanan data pada array untuk mempermudah penampilan data. Proses selanjutnya yaitu penentuan pada saat sensor telah mendeteksi 4x lap dari object (mobil), maka dengan otomatis sensor tidak akan menghitung lagi (meski ada object yang terdeteksi lagi).

Main.c

Pada program main terdapat program untuk timer. Dimana timer terdiri atas jam menit dan detik yang ditampilkan pada LCD. Berikut potongan programnya :

```
void HAL_TIM_PeriodElapsedCallback(TIM_HandleTypeDef *htim2)
{
    if(htim2->Instance== TIM2)
    {
        iddle=1;
        tikupdate();
        if(start==1){
            ms100++;
            if(ms100>=10){
                detik++;
                if(detik>=60){
                    detik=0;
                    menit++;
                }
                ms100=0;
            }
        }
        sprintf(lcd,"%d:%d:%d",menit,detik,ms100);
    }
}
```

Pada main juga terdapat program sensor deteksi yang terdiri dari start, reset, dan sensor 1 sampai 3. Berikut contoh potongan program sensor 2 :

```
if(sensor2Detected()==Normal&&start==1)
{
    mainLap2++;
    lcd_gotoxy(1,0);
    sprintf(kata,"M2 L%d = %d:%d:%d",mainLap2,menit,detik,ms100);
    lcd_putstr(kata);
    HAL_Delay(2000);
    lcd_gotoxy(1,0);
    sprintf(kata,"                 ");
    lcd_putstr(kata);
    HAL_Delay(10);
    //sprintf(kata,"Mobil 2 -> lap%d = %2d:%2d:%2d\r\n",mainLap1,m
    HAL_UART_Transmit(&huart1,kata,28,10);
    HAL_GPIO_TogglePin(GPIOC,LEDMERAH_Pin);
    lcd_gotoxy(0,12);
    sprintf(kata,"%d",mainLap2);
    lcd_putstr(kata);
    if(mainLap2>=4)
    {
        //sprintf(kata,"Mobil 2 -> FINISHED\r\n",mainLap1,menit,de
        lcd_gotoxy(1,0);
        lcd_putstr("Car 2->FINISHED");
        HAL_Delay(2000);
        lcd_gotoxy(1,0);
        lcd_putstr("                ");
        HAL_Delay(10);
        HAL_UART_Transmit(&huart1,kata,21,10);
        mainLap2=0;
        lcd_gotoxy(0,12);
        sprintf(kata,"%d",mainLap2);
        lcd_putstr(kata);
    }
}
else if(sensor2Detected()==Error&&start==1)
{
    sprintf(kata,"SENSOR 2 error\r\n");
    HAL_UART_Transmit(&huart1,kata,16,10);
    lcd_gotoxy(1,0);
    lcd_putstr("s 2 error");
    HAL_Delay(1500);
    lcd_gotoxy(1,0);
    lcd_putstr("                ");
    HAL_Delay(10);
}
```

Dari program diatas terlihat bahwa. Saat sensor 2 mendeteksi, mainLap akan bertambah dan kondisi lap dan waktu akan ditampilkan pada LCD. Apabila mainLap lebih sudah menunjukkan angka 4, maka mobil sudah finish yang di deklarasi pada LCD dengan tulisan “Car 2->FINISHED”, lalu mainLap akan kembali ke kondisi 0. Lalu saat apabila sensor 2 error, maka akan masuk ke kondisi program timeout.

Sensor.c

Pada program sensor ini terdapat 5 sensor yaitu, sensor 1, sensor 2, sensor 3, Reset, dan Start. Dimana program sensor akan dijelaskan berikut. Berikut potongan salah satu programnya :

```
uint8_t sensor2Detected(void){
    int hasil=0;
    int timeOut;
    if(HAL_GPIO_ReadPin(Sensor2_GPIO_Port,Sensor2_Pin)== GPIO_PIN_RESET){
        HAL_Delay(40);
        if(HAL_GPIO_ReadPin(Sensor2_GPIO_Port,Sensor2_Pin)==GPIO_PIN_RESET)  {
            timeOut = tik;
            while(HAL_GPIO_ReadPin(Sensor2_GPIO_Port,Sensor2_Pin)==GPIO_PIN_RESET&&(tik-timeOut)<counterTimeout);
            if(tik-timeOut<counterTimeout)hasil=1; //kode normal
            else{ hasil=5; #define counterTimeout   30
            }
        }
    }
    return(hasil);
```

Pada program diatas, saat sensor mendeteksi, akan di didelay 40ms. Adanya delay ini untuk menghilangkan bouncing pada sensor saat perubahan logika 0 ke 1. Setelah itu, sensor ditanyakan lagi untuk pendeteksiannya untuk menghilangkan spike pada sensor. Penggunakaan program anti bouncing dan spike ini yang diaplikasikan pada semua sensor.

Lcd.c

Pada program LCD ini untuk mengaktifkan fungsi program LCD pada sistem embedded ini. Dimana pada LCD terdapat pin EN, RS, RW, D4, D5, D6, dan D7 yang akan diaktifkan pada pin GPIO ARM STM32F1. Berikut potongan programnya :

```
#include "stm32f1xx_hal.h"
#include "device.h"
#include "lcd.h"
void enable()
{
    HAL_GPIO_WritePin(GPIOB,EN_Pin,GPIO_PIN_SET);//set en
    HAL_Delay(3);    // minimal 2ms don't follow databook ;((450ns)<--
    HAL_GPIO_WritePin(GPIOB,EN_Pin,GPIO_PIN_RESET);//clear en
}
void lcd_init()
{
    HAL_GPIO_WritePin(GPIOB,EN_Pin,GPIO_PIN_RESET);
    HAL_Delay(20);
     //step1
    PORT(0x30);
    HAL_GPIO_WritePin(GPIOB,RS_Pin|RW_Pin,GPIO_PIN_RESET);enable();
    HAL_Delay(5);
     // step2
    PORT(0x30);
    HAL_GPIO_WritePin(GPIOB,RS_Pin|RW_Pin,GPIO_PIN_RESET);enable();
    HAL_Delay(1);
    // step3
    PORT(0x30);
    HAL_GPIO_WritePin(GPIOB,RS_Pin|RW_Pin,GPIO_PIN_RESET);enable();
    //next init
    PORT(0x20);
    HAL_GPIO_WritePin(GPIOB,RS_Pin|RW_Pin,GPIO_PIN_RESET);enable();
    PORT(0x20);
    HAL_GPIO_WritePin(GPIOB,RS_Pin|RW_Pin,GPIO_PIN_RESET);enable();
    PORT(0x80);
    HAL_GPIO_WritePin(GPIOB,RS_Pin|RW_Pin,GPIO_PIN_RESET);enable();
    PORT(0x00);
    HAL_GPIO_WritePin(GPIOB,RS_Pin|RW_Pin,GPIO_PIN_RESET);enable();//
    PORT(0xE0);
```

Uart.c

```
#include "uart.h"
#include "stm32f1xx_hal.h"
UART_HandleTypeDef huart1;
void MX_USART1_UART_Init(void)
{
    GPIO_InitTypeDef GPIO_InitStruct;
        __HAL_RCC_USART1_CLK_ENABLE();
        /**USART2 GPIO Configuration
            PA2     ------> USART2_TX
            PA3     ------> USART2_RX
         */
        GPIO_InitStruct.Pin = GPIO_PIN_9|GPIO_PIN_10;
        GPIO_InitStruct.Mode = GPIO_MODE_AF_PP;
        GPIO_InitStruct.Pull = GPIO_PULLUP;
        //GPIO_InitStruct.Speed = GPIO_SPEED_FREQ_VERY_HIGH;
        HAL_GPIO_Init(GPIOA, &GPIO_InitStruct);

  huart1.Instance = USART1;
  huart1.Init.BaudRate = 9600;
  huart1.Init.WordLength = UART_WORDLENGTH_8B;
  huart1.Init.StopBits = UART_STOPBITS_1;
  huart1.Init.Parity = UART_PARITY_NONE;
  huart1.Init.Mode = UART_MODE_TX_RX;
  huart1.Init.HwFlowCtl = UART_HWCONTROL_NONE;
  huart1.Init.OverSampling = UART_OVERSAMPLING_16;
  if (HAL_UART_Init(&huart1) != HAL_OK)
  {
    _Error_Handler(__FILE__, __LINE__);
  }
}
void print_UART1(uint8_t string[], uint8_t length){
    HAL_UART_Transmit(&huart1, (uint8_t*)string, length, 5);
}
```

Pada program diatas merupakan program UART. Dimana pada UART pin yang digunakan adalah pin A9 dan A10. Pin A9 sebagai RX dan pin A10 sebagai TX. Untuk baudrate digunakan pada 9600. Data 8 bit, mode Normal, dan parity 1. Untuk format pengiriman serial menggunakan program

HAL_UART_Transmit(&huart1, (uint8_t)string, length, 5);*.

Hardwareinit.c

Pada program ini yaitu untuk mendeklarasikan kondisi pin pin yang digunakan seperti reset, start, sensor, dan LCD. Berikut potongan programnya :

```
GPIO_InitStruct.Speed = GPIO_SPEED_FREQ_LOW;
HAL_GPIO_Init(LEDMERAH_GPIO_Port, &GPIO_InitStruct);

/*Configure GPIO pins : pbUp_Pin pcRight_Pin */
GPIO_InitStruct.Pin = Sensor1_Pin|Sensor2_Pin;
GPIO_InitStruct.Mode = GPIO_MODE_INPUT;
GPIO_InitStruct.Pull = GPIO_PULLUP;
HAL_GPIO_Init(GPIOC, &GPIO_InitStruct);

GPIO_InitStruct.Pin = Sensor3_Pin;
GPIO_InitStruct.Mode = GPIO_MODE_INPUT;
GPIO_InitStruct.Pull = GPIO_PULLUP;
HAL_GPIO_Init(GPIOA, &GPIO_InitStruct);
/*Configure GPIO pins : Start_Pin Reset_Pin */
GPIO_InitStruct.Pin = Reset_Pin;
GPIO_InitStruct.Mode = GPIO_MODE_INPUT;
GPIO_InitStruct.Pull = GPIO_NOPULL;
HAL_GPIO_Init(GPIOB, &GPIO_InitStruct);
GPIO_InitStruct.Pin = Start_Pin;
GPIO_InitStruct.Mode = GPIO_MODE_INPUT;
GPIO_InitStruct.Pull = GPIO_NOPULL;
HAL_GPIO_Init(GPIOA, &GPIO_InitStruct);

GPIO_InitStruct.Pin = D7_Pin|D6_Pin|D5_Pin|D4_Pin|RS_Pin|RW_Pin|EN_Pin;
GPIO_InitStruct.Mode = GPIO_MODE_OUTPUT_PP;
GPIO_InitStruct.Pull = GPIO_NOPULL;
GPIO_InitStruct.Speed = GPIO_SPEED_FREQ_LOW;
HAL_GPIO_Init(GPIOB, &GPIO_InitStruct);

}
```

Pada program diatas, keadaan sensor 1 sampai sensor 3 di mode sebagai input. Dan keadaan awal pin-pin sensor dalam keadaan pullup. Begitu juga pada pin Reset dan Start di mode sebagai input dan keadaan pullup. Untuk pin LCD di mode sebagai output namun kondisi Pulldown.

device.h

pada program ini dimana di deklarasi pin pin yang akan digunakan sesuai kebutuhan input dan output. Berikut program nya :

```
#ifndef DEVICE_H_
#define DEVICE_H_
#define D7_Pin GPIO_PIN_3
#define D7_GPIO_Port GPIOB
#define D6_Pin GPIO_PIN_4
#define D6_GPIO_Port GPIOB
#define D5_Pin GPIO_PIN_5
#define D5_GPIO_Port GPIOB
#define D4_Pin GPIO_PIN_6
#define D4_GPIO_Port GPIOB
#define RS_Pin GPIO_PIN_9
#define RS_GPIO_Port GPIOB
#define RW_Pin GPIO_PIN_8
#define RW_GPIO_Port GPIOB
#define EN_Pin GPIO_PIN_7
#define EN_GPIO_Port GPIOB
#define Sensor3_Pin GPIO_PIN_15//2
#define Sensor3_GPIO_Port GPIOC//B
#define Sensor2_Pin GPIO_PIN_14//2
#define Sensor2_GPIO_Port GPIOC//A
#define Sensor1_Pin GPIO_PIN_13//4
#define Sensor1_GPIO_Port GPIOC//B
#define Buzzer_Pin GPIO_PIN_11
#define Buzzer_GPIO_Port GPIOC//A
#define Start_Pin GPIO_PIN_12//5
#define Start_GPIO_Port GPIOB
#define Reset_Pin GPIO_PIN_11//8
#define Reset_GPIO_Port GPIOA
```

Dari program diatas terlihat bahwa pin LCD yaitu D7 => B3, D6 => B4, D5 => B5, D4 => B6, EN => B7, RW => B8, dan RS => B9. Lalu pada sesnor 1 pada pin C 13,

pada sensor 2 pada pin C14, dan sensor 3 pada pin C15. Untuk tombol start pada pin B12 dan Reset pada pin A11.

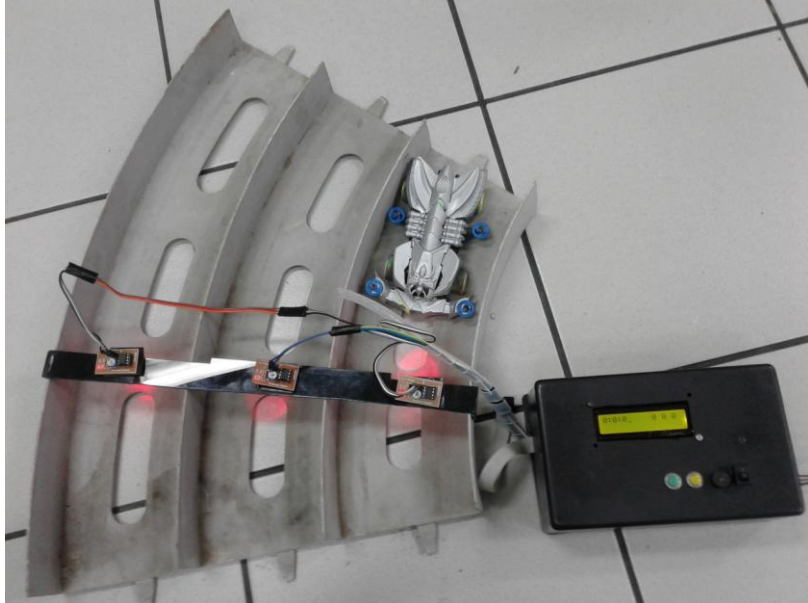
Kesimpulan

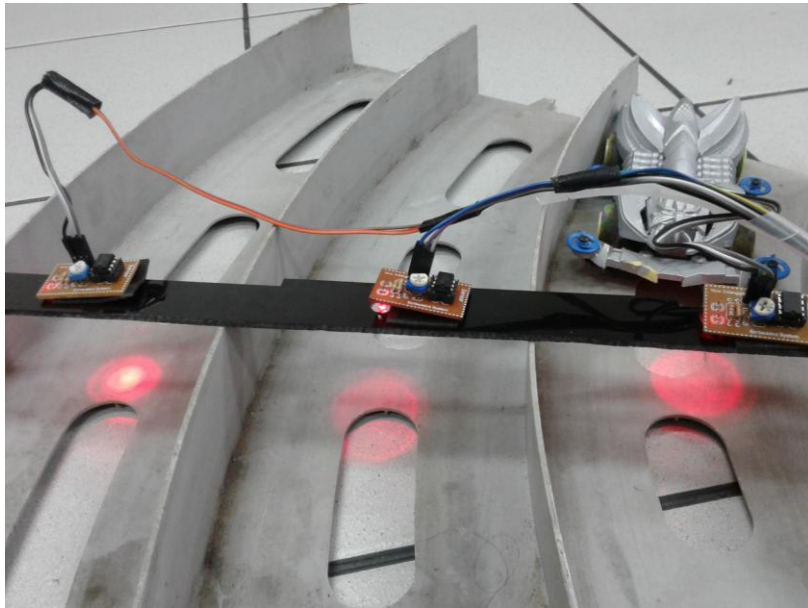
Dari hasil penjelasan diatas dapat di simpulkan bahwa :

1. Timer pada ARM dapat digunakan sebagai pewaktuan yang sebenarnya dengan menggunkan sumber clock internal
2. Embedded Controller dapat mengirimkan data serial dengan menggunkan UART
3. Mekanisme bouncing dilakukan untuk menghindari bouncing yang terjadi pada setiap sensor dan push button.
4. Mekanisme timeout dilakukan untuk mengetahui sensor mana yang tidak berfungsi
5. Mekanisme managing program dilakukan untuk mempermudah dalam pembuatan program.
6. Lcd dapat menampilkan data hasil proses pada tampilannya.

Lampiran

Hardware

- **Car Lap Counting**
  Pemasangan pada lintasan

- **Box Mikrokontroller**
  Didalam box ini terdapat board shield ARM STM32F1, Sistem Supplybeserta pengisian, dan Serial.

- **Sensor**
  Terdapat 3 sensor yang di package sedemikian hingga, sehingga cukup dipasang pada lintasan.

Tampilan LCD

- **Sensor 1 Mendeteksi**

- **Sensor 1 Error**

- **Sensor 2 Mendeteksi**

- **Sensor 2 Error**

- **Sensor 3 Mendeteksi**

- **Sensor 3 Error**

- **Reset**